two bit stack pointer, the sixty-four bit procedure responds to the same underflow trap without any change to the procedure other than the response to restore from different save positions once a sixty-four bit procedure has been detected by the trap.

Since the information in the stack pointer must be used by the procedure, if the procedure is a sixty-four bit procedure, the system leaves the stack pointer in the upper area in its original condition with a one in the lowest bit position so that no time is lost when the register file is restored. As is also illustrated in FIG. 3, the one remains in bit position 63 designating a sixty-four bit procedure even though the lowest order bit of the stack pointer in the thirty-two bit save area provides the same information while the register file is stored in the save area.

It should be noted that one problem which exists with the use of the same sixty-four bit registers for both thirty-two and sixty-four bit procedures is that a register file executing a thirty-two bit procedure may include a number of higher order bits which are not related to the procedure. In some cases these bits can adversely affect the outcome of the procedure being run. For example, if the register contains an address to be used in a load or a store operation for a thirty-two bit procedure, if the upper bits are used, they produce the wrong address. Consequently, these bits must be masked off in some manner. In the preferred embodiment of the present invention, a mask register is used to store information relating to the particular bits of the address which are to be used. When data is loaded or stored using an address held in the register file, the mask register is used to force the state of the upper bits which would otherwise carry invalid information. The details of such a mask register are described in more detail in U.S. Pat. No. 5,210,839, entitled METHOD AND APPARATUS FOR PROVIDING A MEMORY ADDRESS FROM A COMPUTER INSTRUCTION USING A MASK REGISTER, Powell et al, filed on even date herewith and assigned to the assignee of the present invention.

FIGS. 4*a* and 4*b* illustrate and reiterate the steps used in practicing the method of this invention.

Although the present invention has been described in terms of a preferred embodiment, it will be appreciated that various modifications and alterations might be made by those skilled in the art without departing from the spirit and scope of the invention. The invention should therefore be measured in terms of the claims which follow.

What is claimed is:

1. A method for context switching a processor that executes procedures having differing word sizes, comprising the steps of:

testing a most significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the most significant bit of the stack pointer register indicates the first word size;

setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the most significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

2. The method of claim 1, further comprising the steps of:

testing the width indication bit in the first stack save area in memory;

reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the most significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

3. The method of claim 2, wherein the width indication bit in the first stack 'save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

4. The method of claim 2, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

5. The method of claim 4, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

6. The method of claim 5, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

7. The method of claim 6, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

8. The method of claim 7, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.

9. A processor that executes procedures having differing word sizes, comprising:

means for testing a most significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

means for transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the most significant bit of the stack pointer register indicates the first word size;

means for setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the most significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

10. The processor of claim 9, further comprising:

means for testing the width indication bit in the first stack save area in memory;

means for reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the most significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

means for reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in mem ry indicates that the data values for the prócedure have the second word size.

11. The processor of claim 10, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

12. The processor of claim 10, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

13. The processor of claim 12, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

14. The processor of claim 13, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

15. The processor of claim 14, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

16. The processor of claim 15, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.

* * * * *

IN THE TITLE

At line 3, change [BIT] to --BIT OR LEAST SIGNIFICANT BIT--.

IN THE CLAIMS

Please add Claims 17-33 as follows:

17. A method for context switching a processor that executes procedures having differing word sizes, comprising the steps of:

testing a least significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the least significant bit of the stack pointer register indicates the first word size;

setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the least significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

18. The method of claim 17, further comprising the steps of:

testing the width indication bit in the first stack save area in memory;

reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the least significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

19. The method of claim 18, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

20. The method of claim 18, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

21. The method of claim 20, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

22. The method of claim 21, wherein the first word size comprises **32** bits and the second word size comprises **64** bits.

23. The method of claim 22, wherein the registers in the processor and the stack pointer register in the processor comprise **16** registers each comprising **64** bits.

24. The method of claim 23, wherein the offset value and the area in memory for the first stack save area each comprise **16** multiplied by **4** bytes per register.

25. A processor that executes procedures having differing word sizes, comprising:

means for testing a least significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

means for transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the least significant bit of the stack pointer register indicates the first word size;

means for setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the least significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

26. The processor of claim 25, further comprising:

means for testing the width indication bit in the first stack save area in memory;

means for reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the least significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

means for reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

27. The processor of claim 26, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

28. The processor of claim 26, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

29. The processor of claim 28, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

30. The processor of claim 29, wherein the first word size comprises **32** bits and the second word size comprises **64** bits.

31. The processor of claim 30, wherein the registers in the processor and the stack pointer register in the processor comprise **16** registers each comprising **64** bits.

32. The processor of claim 31, wherein the offset value and the area in memory for the first stack save area each comprise **16** multiplied by **4** bytes per register.

Sub B1

33. ~~A method for context switching a processor that executes~~ procedures having differing word sizes, comprising the steps of:

testing a least significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor, each have a first word size or a second word size, wherein the first word size is less than the second word size;

transferring the data values from a least significant portion of each register to a first stack save area in memory if the least significant bit of the stack pointer register indicates the first word size;

transferring the data values from the registers to a second stack save area in memory if the least significant bit of the stack pointer register indicates the second word size;

setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the least significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates ~~that the data values for the procedure have the second word size.~~

Add A1